

Customized Edit Interfaces for Wikis via Semantic Annotations

Angelo Di Iorio¹, Silvia Duca¹, Alberto Musetti¹, Silvia Righini¹, Davide Rossi¹,
Fabio Vitali¹

¹ Department of Computer Science, University of Bologna, Mura Anteo Zamboni 7,
40127 Bologna, Italy

{ diorio@cs.unibo.it, silvia.duca@unibo.it, musetti@cs.unibo.it, righini@cs.unibo.it,
rossi@cs.unibo.it, fabio@cs.unibo.it }

Abstract.

Authoring support for semantic annotations represent the wiki way of the Semantic Web, ultimately leading to the wiki version of the Semantic Web's eternal dilemma: why should authors correctly annotate their content? The obvious solution is to make the ratio between the needed effort and the acquired advantages as small as possible. Two are, at least, the specificities that set wikis apart from other Web-accessible content in this respect: social aspects (wikis are often the expression of a community) and technical issues (wikis are edited "on-line"). Being related to a community, wikis are intrinsically associated to the model of knowledge of that community, making the relation between wiki content and ontologies the result of a natural process. Being edited on-line, wikis can benefit from a synergy of Web technologies that support all the information sharing process, from authoring to delivery.

In this paper we present an approach to reduce the authoring effort by providing ontology-based tools to integrate models of knowledge with authoring-support technologies, using a functional approach to content fragment creation that plays nicely with the "wiki way" of managing information.

Keywords: mediawiki, metadata, editor, template, semantic, Web 2.0

1. Introduction

As a first approximation, editing a wiki page is a very easy thing to do: we find the page that needs new content and click on the "Edit" button; either a WYSIWYG editor or a text box (if we are unlucky) are shown and off we go with writing down our stream of content. Then we click on the "Save" button and the page is shown updated.

This approach works perfectly as long as the content of the page is made of free-flow text and no structured data, so that the act of writing it resembles as much as possible the kind of writings possible on traditional desktop word processors.

Wiki content is often expected to assume a more precise organization: e.g. a given structure, the presence of one or more predetermined elements, a specific sequence of parts, a given hierarchical organization, etc.. Some wiki applications have introduced the concept of template to help the page author to remember all the appropriate parts and the expected structure of the page. These templates are usually associated to page categories (chosen by the content author at the beginning of the editing process) and can be classified in two categories, called *creational* and *functional* [3]:

1. *creational* templates create new whole pages as a copy of an existing page, adding expected fragments (e.g., section headings) and boilerplate text to be substituted by the content author (e.g.: the words “insert description of page here”),
2. *functional* templates create a parameterized environment where named procedures can be invoked with actual parameter values to generate a completely different content fragment, such as infoboxes in Mediawiki² that use a functional syntax in the edit page to generate a structured name-value table in the rendered page.

In previous papers [2] and [7] we have shown how light constraints and user-defined rule-based annotations can help authors to verify the content correctness of a page, after the editing action. Such a process does not impose an excessive burden in actions, requirements and controls that would spoil or ruin the easy-going and ultimately successful “wiki way” of editing content [1].

In this paper we examine how content authors can be further helped in generating structured content in a manner that is compatible, again, with the aforementioned “wiki way” of editing.

Our approach is based on Semantic Web technologies to automatically generate either forms or templates. These interfaces are expressively equivalent but structurally independent of the expected content of the rendered page. Users do not deal with wiki code or complex syntaxes, rather they manipulate consistent (semantic) data in a transparent way.

More important, such an automatic process produces customized interfaces that can be easily tailored for each user or class of users. Authors use their own interfaces and can select, organize and customize their content with little effort.

The core of our proposal is the exploitation of ontologies for describing both customized interfaces and content. Ontologies are used to provide an abstract description of the concepts underlying the wiki page, and yet are separated from the details of the templates and forms, described through a separate ontology.

In this paper we introduce two separate but complementary tools, called Gaffform and TinPP, which can be used to enrich MediaWiki with ontology-based functionality for the editing and presentation of structured data.

¹ Throughout the paper we will often use the term “expect” whenever we are tempted to use (or some readers would not be surprised to find) the term “require” to refer to the behavior of wiki authors and readers. This is to stress the fact that the full adoption of the *wiki way*, i.e., the philosophy of use and content generation introduced by wiki software, is generally speaking not compatible with impositions, rules and constraints on final users, and adherence to such rules should always be considered as voluntary and optional rather than imposed by the software.

² <http://www.mediawiki.org/>

Gaffeform uses a three-ontology model to generate arbitrary and customized forms for the editing of classes and class properties within MediaWiki. TinPP uses the same three ontologies to generate MediaWiki templates that can be used for rendering properties within the wiki page and along with the free-flow text of the same page. Together, they can be used to generate custom editing and rendering interfaces for MediaWiki functional templates. The end result is to generate structured content without ever forcing the user to adopt a given template nor to learn how to edit MediaWiki templates.

The paper is structured as follows: in section 2 we describe existing approaches at using semantic Web technologies in wikis. In section 3 we discuss the issues connected to generating custom interfaces through a three-ontology model, and in section 4 we introduce Gaffeform and TinPP, the tools generated to showcase our approach.

2. Related works

Combining wikis and semantic technologies is not a new idea in the literature. Semantic wikis are enhanced wikis that allow users to decorate pages with semantic data and to create a shared knowledge-base. Semantic wikis provide users with sophisticated searching and analysis facilities, keeping the open editing philosophy that characterizes traditional wikis.

Some semantic wikis allow users to load a predefined ontology in the wiki platform and automatically create pre-populated pages, that users can further modify. Makna [4] adopts such an approach and produces advanced wiki pages for the manipulation of the concepts of the ontology. While Makna is a complete wiki clone, WikiFactory[8] is a server-side application for the automatic transformation of OWL ontologies into wiki pages, deployable on multiple wiki platforms. Both these projects rely on the strong distinction between roles: the content-author knows the wiki domain but does not have any technical skills, while an ontology expert masters OWL-related tools to map ontology concepts into wiki pages. The system is in charge of translating such an abstract description into actual pages.

Kawawiki [5, 6] extends such an approach providing to the users an editing environment to customize and configure the final pages. Kawawiki takes an input RDF templates – a simplification of the RDF full language – describing both domain concepts and wiki instances. More important, Kawawiki automatically produces forms for populating the wiki pages after their initial deployment. These forms are described in the input RDF templates file and can be further customized by users. However their customization requires users to master a quite complex syntax.

Similarly, SemanticForms [11] *automatically* generates forms for letting users insert structured data in wiki pages. Semantic Forms is an extension of MediaWiki, as the tools presented in this paper. Semantic Forms does not take as input an OWL/RDF file but a MediaWiki template source code. It works on the top of structured content, that could also be created through automatic processes, to provide users a simplified interface for the authoring and customization of content.

WikiTemplate [10] is a closely related project, that allows users to access a page in two modes: when a page is viewed it is formatted according to its view template; when it is edited a set of editable text area is supplied, each corresponding to an area of the template. Common users can freely edit each field of that form, while *tailors* can combine different pieces into complex forms that better match both the structure of the content and the practices of the community.

The idea of combining atomic assets to generate customized interfaces is actually rooted in the early days of the Computer Supported Cooperative Work research. ObjectsLens [9] and its predecessor Information Lens [12] aimed at providing users with an environment for building their own collaborative applications. They adopted a modular approach: users could customize and combine atomic building blocks (forms, input fields, tables, messages) without following pre-constrained paths. The systems were, in fact, characterized as semi-formal systems, “processing some information in formally specified ways and other information in ways that are not formally specified”. A very similar approach will be later adopted by the aforementioned wiki-related tools to provide users simplified authoring interfaces, without twisting and forcing the (semantic) structure of the content.

3. Generating Editing Interfaces from OWL ontologies

The creation of semantic content for wikis is still an open issue. Even more difficult is the creation of personalized workflows that allow users to create content on the basis of their preferences **and skills**. In fact, most of the applications described in the previous section assume that users know how to master languages like OWL and/or tools like Protégé [13]. That solution allows designers to create powerful semantic wikis but it does not solve issues related to authoring and usability.

This paper proposes a different approach, aiming at simplify the production of structured data by providing users with *customized interfaces*. The key idea is to let wiki administrators personalize forms and templates through a very simple and dynamic interface and to automatically produce end-users ‘tools’ that will actually allow authors to insert new semantic data.

The first relevant aspect of our approach is the strong distinction between roles involved in the creation of structured content. In particular, we identified three different actors with very different skills and very different tasks to complete:

1. **Anna:** the final wiki author. She writes the content and add metadata. She knows the topic of the wiki and the content of the pages. On the other hand, she does not have any technical expertise on semantic technologies or Web languages.
2. **Andrea:** the interface and usability expert. He knows how to map domain concepts to interface widgets and organize forms and templates. On the other hand, he does not have any knowledge of semantic technologies. Moreover, his tasks are independent from the actual content of the wiki.

3. **Pino:** the semantic expert. He knows tools and technologies related to the Semantic Web. He is not, however, an expert of the content domain, nor expert of Web technologies.

Note that the same user can play different roles at different times. In fact, each role represents a given category of skills (and goals) rather than a physical person accessing the wiki.

The contribution of this work is the design and implementation of a framework for simplifying the authoring and customization process of semantic data *by exploiting different capabilities of these roles*. The architecture and workflow we propose is shown in figure 1.

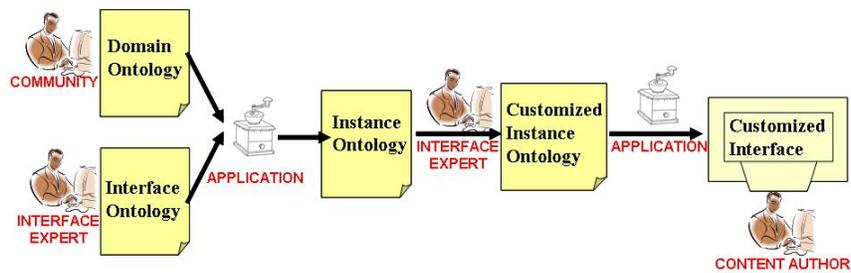


Figure 1: the overall architecture.

The framework takes as input two different ontologies - respectively called *Domain Ontology* and *Interface Ontology* – and automatically produces an intermediate ontology, the *Instance ontology*, describing a basic interface to handle structured data for the input domain. This ontology is transformed into a *Customized instance ontology*, that is finally instantiated into the actual interface shown to the users. Let us briefly discuss each of these ontologies, along with the process for creating them and their usage.

The *Domain Ontology* is actually created by Pino, reflecting ideas and comments of the users about the domain the wiki is related to. This ontology will finally describe properties of each page and relations among them, as domain entities will be mapped into actual wiki entities. There is no pre-defined structure for this ontology, that can be used to model any domain.

The *Interface Ontology* describes graphical interfaces. In particular, it describes objects like text areas, text fields, buttons or checkboxes that can be composed into complex interfaces. Constraints and relations among these objects can also be expressed with the ontology, as well as simple static interfaces. This ontology is created by Andrea though a graphic environment or with the help of Pino, the expert of semantic languages and tools. Basically, such an ontology describes the capabilities of the system and the potential interfaces users can use.

The *Instance Ontology* is automatically created by an application that merges the domain and interfaces ontologies. Domain-related entities are transformed into interfaces widgets that allow users to insert actual data. These widgets might be simple text areas whose content is written in the wiki template syntax, as well as sophisticated and dynamic tools.

Although such a description is enough to automatically generate the final interface, expert users (Andrea) can further customize it. This process is manual, possibly through ad-hoc tools. What is important is that the complexity of managing ontologies is hidden to Andrea, that only masters high level concepts related to interfaces and forms. The result of such a customization is a new ontology, just called *customized instance ontology*, describing the final refined interface.

The last step of the process is completely automatic and consists of mapping the abstract description into an actual template editing interface. Such an instantiation is performed by the system aggregating atomic widgets initially described in the interface ontology. One of the most important aspects of this approach is its *modularity*. Each actor - a human user or a system module - has to perform a specific and well-defined task. In particular, content authors are not required to master semantic tools or Web technologies: their input is filtered into ontology descriptions instantiated *automatically* in the wiki. A second advantage is the independence between the interface to modify the content and the content itself: multiple interfaces, in fact, can be created from the same domain ontology. Different users with different skills and preferences can then customize their interfaces to manipulate shared knowledge-bases. The declarative approach is a final important aspect: modifications to the interface are applied by changing their abstract description and translated into the final wiki without writing any line of code. Changes and updates are automatically propagated without any further intervention of the developers. All these points contribute to make such an approach a good solution to simplify the authoring and customization process of wiki templates, without burdening users with complex structures and tasks. In fact, the complexity of the architecture exists but it is hidden to the final users that only work according to their own skills.

4. From model to implementation: Gaffeform and TinPP

Two prototypes, called Gaffeform and TinPP, have been implemented following the aforementioned ontology-based approach. They are both presented in this section. Gaffeform is an Ajax-based editing environment for customizing forms within Semantic Media Wiki³. TinPP is a Java application for automatic generation of wiki templates, that provided preliminary studies and results for the overall Gaffeform framework.

4.1. TinPP

TinPP is a Java library, developed as a MediaWiki extension, for automatic generation of templates from OWL ontologies. The basic idea is to automatically map ontology *classes and properties* into wiki *categories and fields* of MediaWiki templates. In terms of the general architecture described in the previous section, TinPP takes as input a Domain Ontology, created by Pino on the top of multiple inputs from the domain experts, and a very simple Interface Ontology that describes

³ <http://semantic-mediawiki.org/>

the final interface as a plain wiki textarea, supposed to be manually modified by the content authors. Even the customization process – the generation of the Customized Interface Ontology – is very simple as it only consists of changing the order of fields in the template.

In fact, the focus of TinPP was not to provide users a sophisticated customization environment such as the complete Gaffeform infrastructure. Rather, to test the automatic generation of ‘wiki data’ from OWL ontologies and the extensibility of our approach. The results were very promising and allowed us to create a library later integrated in the overall framework.

The actual output is a wiki template, written in the MediaWiki syntax. The TinPP module saves the template directly in the wiki installation, so that the new template is available to any user about to start a new page.

Another interesting aspect of TinPP is the support for both *forced* and *unforced* template editing. The template creation process can be configured to force a choice between available categories and, ultimately, ontological classes. Or, it can be left to the community the possibility of suggesting specific templates or not using templates at all.

4.2. Gaffeform

Gaffeform is an integrated system that implements the general architecture described in section 3. It provides users with a dynamic environment to create structured wiki content in a simple and personalized way. In particular, it allows users to create forms from OWL ontologies, to customize these forms and to insert actual data through these customized and user-friendly interfaces.

The application is composed of three modules, delivered with different technologies:

- a *MediaWiki extension*: a module integrated in MediaWiki, and written in PHP, that adds Gaffeform functionalities to pages and interfaces of the basic installation;
- an *Ontology manager*: a module that processes OWL ontologies to produce final forms. Actually, this manager uses internally the Jena OWL processor. It is a Web-service, written in Java, that provides ontology-related tools to the MediaWiki extension;
- an *Ajax-based interface*: a client-side module that allows users to actually insert data through the forms generated by the Gaffeform engine.

The workflow integrating these three modules follows the schema described so far. The MediaWiki extension provides users an interface to upload the Domain Ontology and the Interface Ontology: these two operations are actually presented in different panels and are available to different users.

The input ontologies are automatically transformed into a simple and general form, by aggregating some widgets pre-loaded in the platform.

The first time a user edits a page, the MediaWiki extension shows such a basic form. The author can then organize a new form adding dynamic behaviors, moving buttons, changing fields order and so on. Modifying the structure and the parameters of the ontologies, the author is able to customize and change any detail of the interface. Figure 2 shows a basic form and a possible personalization.

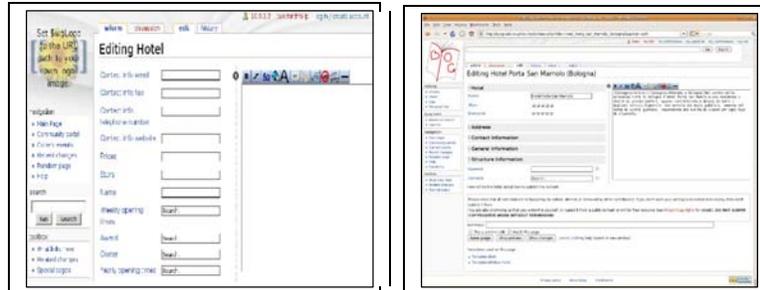


Figure 2: a general and personalized forms produced by Gaffeform

The current implementation does not provide users any simple interface to modify the ontologies but it still require the manual modification of these files. We just started to implement this new module that completes the architecture.

What is important is that any form is generated *automatically* by Gaffeform, integrating Javascript widgets and form objects described in the Interface Ontology. There is no manual generation of code but users adopts a declarative approach to aggregate new forms.

The forms produced by Gaffeform allow final users to produce both plain wiki templates and semantic data – in line with the domain ontology - to be used for advanced searching and analysis. Figure 3 shows a final page with a template (infobox in the right side of the page), whose data were submitted through the forms shown in figure 2.



Figure 3: The final wiki page with data inserted though the customized form

The current implementation of Gaffeform is a Web 2.0 application that simplifies the creation of MediaWiki templates. On the other hand, it still requires a manual intervention on the Interface Ontology. The next step of our research will be the implementation of a graphic tool to customize forms in a simple and fast way, using widgets that will be integrated in the Interface Ontology as well.

A systematic users evaluation of Gaffeform is another very important item in our agenda. Preliminary results showed us that customized and automatically-generated interfaces help users in creating semantic content. We plan to perform larger experiments and evaluations, and to study different trends for different domains and classes of users.

5. Conclusions

In this paper we have presented two separated technologies to help content authors in generating structured content in wiki pages in an easy way through the use of fully customizable form interfaces based on Semantic Web technologies. Although the application of semantic wiki technologies in the field of wikis is not new, the idea of associating them to template editing and to the customization of the interface for the editing of structured information is unique and represents a relevant innovation to the field of interface adaptation. We believe that both Gaffeform and TinPP represent useful approaches for extending the ease of use of wiki editing to structured content as well, and custom interfaces through the use of a three layer ontology can be exploited in a number of different directions, too.

References

1. Cunningham, W. and Leuf, B.. The Wiki way . Addison-Wesley, New York, 2001.
2. Di Iorio A., Zacchiroli S. "Constrained Wiki: an Oxymoron?". In the Proceedings of the ACM Symposium on Wikis 2006, ACM Press, Odense, Denmark, 2006, pp. 89-98.
3. Di Iorio A., Vitali, F., Zacchiroli S., "Wiki Content Templating" in the Proceedings of the World Wide Web Conference 2008, Beijing, China, 2008.
4. Dello, K., Paslaru, E.. Creating and using semantic web information with makna. Proceedings of the First Workshop on Semantic Wikis, Max Völkel and Sebastian Schaffert Editors, 2006.
5. Kawamoto, K., and Kitamura, Y., Tijerino, Y. Kawawiki: A semantic wiki based on RDF template. Proceedings of the 2006 IEEE/WIC/ACM international conference on Web Intelligence and Intelligent Agent Technology, Hong Kong, China - 18-22 Decembre 2006.
6. Kawamoto, K., and Kitamura, Y., Tijerino, Y. Kawawiki: A template-based semanticwiki where end and expert users collaborate. Proceeding of the 5th International Semantic Web Conference, Athens, GA, USA - 5-9 November 2006.
7. Di Iorio A., Rossi D., Vitali F. and Zacchiroli S. "Where are your Manners? Sharing Best Community Practices in the Web 2.0". In Proceedings of ACM SAC 2009 (the 24th Annual ACM Symposium on Applied Computing), Track on Web Technologies.
8. Di Iorio, A., Presutti, V. and Vitali, F. WikiFactory: an ontology-based application to deploy domain-oriented wikis. In Proceedings of the European Semantic Web Conference, Budva, Montenegro, 2006.
9. Lai, K.-Y., Malone, T. W., and Yu, K.-C. 1988. Object lens: a spreadsheet for cooperative work. ACM Transactions on Information Systems 6, 4, 332{353}.
10. Haake, A., Lukosch, S., and Schummer, T. 2005. Wiki-templates: adding structure support to wikis on demand. In WikiSym '05: Proceedings of the 2005 international symposium on Wikis. ACM Press, New York, NY, USA, 41-51.
11. Koren, Y. 2008. Semantic forms: Creating Forms for MediaWiki. http://www.mediawiki.org/wiki/Extension:Semantic_Forms
12. Malone, T. W., Grant, K. R., and Turbak, F. A. 1986. The information lens: an intelligent system for information sharing in organizations. In CHI '86: Proceedings of the SIGCHI conference on Human factors in computing systems. ACM, New York, NY, USA, 1(8).
13. Stanford Center for Biomedical Informatics Research, What is Protégé?, (2007), <http://protege.stanford.edu/overview/index.html>